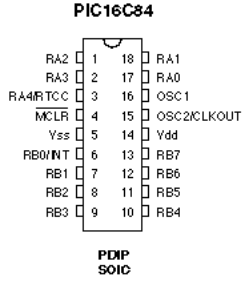
PIC16C84 Pin-Out

The following diagram shows the PIC16C84 pin-out:



Pin	Function
RA0 - RA4	I/O Port A
RB0 - RB7	I/O Port B
INT	External interrupt input
RTCC	Real-time clock/counter input
MCLR	Master clear (reset)
OSC1	Oscillator input
OSC2/CLKOUT	Oscillator output (OSC/4)
Vdd	Power supply
Vss	Ground

PIC16C84

PIC16C84 Microcontrollers

The table below shows the various PIC16C84's available:

Part #	Erasable	Program	Registers	I/O	Power	Osc. Type	Frequency
PIC16C84-04	Yes	1K x 14	100 x 8*	13	4.0 - 6.0 V	RC,XTAL	DC - 4 MHz
PIC16C84-10	Yes	1K x 14	100 x 8*	13	4.0 - 6.0 V	RC,XTAL	DC - 10 MHz
PIC16LC84-04	Yes	1K x 14	100 x 8*	13	2.0 - 6.0 V	RC,XTAL	DC - 4 MHz

* 36 registers in RAM, 64 registers in EEPROM.

New and Modified Features

The PIC16C84 shares many features with the older PIC16C5x and PIC16C71 devices. In fact, with a few notable exceptions, the '84 is a virtual copy of the '71. The following text describes the important differences between the '84 and the original '5x series:

- **1K program space and 64 registers implemented in EEPROM**.
- **Interrupts possible from four sources**: external pin, RTCC timer, EEPROM write complete, and change on four Port B pins.
- **Port B modifications**. Software controlled pull-up's have been added, along with the ability to generate an interrupt when any of four pins changes state.
- **8-level hardware stack**. Allows deeper nesting of subroutines.
- **14-bit instruction word**. Provides larger page sizes for program memory (2K) and RAM (128 bytes).
- **New timers**. Two new timers have been added to control delays on power-up and wake-up. These timers are the oscillator start-up timer (OST) and power-up timer (PWRT).
- **New I/O pin**. An additional I/O pin has been added as bit 4 of Port A. This is physically implemented on pin 3 (RTCC/RA4).
- **Status register changes**. Program page select bits (bit 5-6) have been replaced by *register* page select bits.

New and Modified Features *(continued)*

- **File select register changes**. The FSR has been increased to 8 bits, and bits 5-6 no longer function as register page select (register page selection is now done in the Status register).
- **High byte added to PC**. A high byte has been added to the program counter to handle program memory paging.
- **Page select bits removed**. Program memory page select bits in the Status register (PA0 - PA2) have been removed.

  The part's 1K of program space can be utilized without worrying about page boundaries.
- **Register 07h is unimplemented**. This register cannot be used for storage, as it can in 18-pin PIC16C5x devices.
- **Reset vector moved to 0000h**.
- **Interrupt vector added at 0004h.**
- **Six directives & instructions have been added to the assembler**:

Microchip Instruction	Parallax Equivalent
-	**pwrt_on, pwrt_off** Used in the device directive to control the 84's power-up timer.
sublw	**mov w,#literal-w** Move #literal-w into w.
addlw	**add w,#literal** Add #literal to w.
return	**ret** Return, *without affecting w*.
retfie	**reti** Return from interrupt.
option	**mov !option,data** Load option register with data.

New and Modified Features *(continued)*

- **Five directives & instructions have been removed**:

Microchip Instruction	Parallax Equivalent
-	**reset** In the '84, the reset vector has been moved to address 0000h, so the reset directive was dropped. The reset vector is now set up by establishing a jmp at 0000h to the start of your program; address 0005h is the first possible location for your program.
-	**ljmp, lcall, lset** The '84 does not have paged program memory, so long linstructions are unnecessary.
-	**addb fr,bit** Add bit into file register.
-	**subb fr,bit** Subtract bit from file register.
-	**neg fr** Negate file register.

PIC16C84 Registers

The following table shows the various registers in the PIC16C84; the function of each register is described in the following pages.

Register	Function
-	W register
00h	Indirect addressing register
01h	Real-time clock/counter (RTCC)
02h	Program counter low byte (PCL)
-	Stack registers (8)
03h	Status register
04h	File select register (FSR)
05h	I/O Port A
06h	I/O Port B
07h	Not implemented
08h	EEPROM data register (EEDATA)
09h	EEPROM address register (EEADR)
0Ah	Program counter latch high (PCLATH)
0Bh	Interrupt control register (INTCON)
0Ch - 2Fh	General purpose registers
30h - 7Fh	Not implemented
———	**End of page 0 register memory**
80h	Indirect addressing register
81h	Option register
82h	Program counter low byte (PCL)
83h	Status register
84h	File select register (FSR)
85h	TRISA
86h	TRISB
87h	Not implemented
88h	EEPROM control register 1 (EECON1)
89h	EEPROM control register 2 (EECON2)
8Ah	Program counter latch high (PCLATH)
8Bh	Interrupt control register (INTCON)
8Ch - AFh	Reads/writes registers 0Ch - 2Fh
B0h - FFh	Not implemented

PIC16C84 Registers *(continued)*

Quick reference register map:

	PAGE 0	PAGE 1	
	W register	W register	
	Stack (8)	Stack (8)	
00h	Indirect addr.	Indirect addr.	80h
01h	RTCC	OPTION	81h
02h	PCL	PCL	82h
03h	STATUS	STATUS	83h
04h	FSR	FSR	84h
05h	PORT A	TRISA	85h
06h	PORT B	TRISB	86h
07h			87h
08h	EEDATA	EECON1	88h
09h	EEADR	EECON2	89h
0Ah	PCLATH	PCLATH	8Ah
0Bh	INTCON	INTCON	8Bh
0Ch	36 general purpose registers (RAM)	reads/writes registers 0Ch - 2Fh	8Ch
2Fh			AFh
30h			B0h
7Fh			FFh

PIC16C84 Registers *(continued)*

The following text describes the function of each *new or modified* register. Refer to the PIC16C5x section for registers not shown here.

For some of the registers, you'll notice the designation "xxh" following the register name. This indicates the address of the register. Registers with no address cannot be addressed directly.

- **Program Counter Low Byte (02h: PCL).** The program counter holds the address for the instruction currently being executed. The program counter and its associated eight-level stack are 13 bits wide, with PCL holding the lower 8 bits and PCH holding the upper 5 bits. PCH is automatically loaded when a JMP or CALL is performed. Although PCH is not addressable, your program can load PCH through the PCLATH register (see register f0A, later in this section).
- **Stack.** The stack is comprised of eight registers which are used for calling and returning from subroutines. The program counter is pushed onto the stack when a CALL is executed or an interrupt is acknowledged. The stack is popped in the event of a RET, RETW, or RETI instruction.
- **Status Register (03h).** This register contains the status of the arithmetic logic unit (ALU), the reset status, and the page select bits for *register* memory (not program memory, as in PIC16C5x devices).

  The function of each bit in the status register is shown below:

Bit	Function
0	**Carry bit (C).** Set if an addition or subtraction causes an overflow from the most significant bit of the resultant (bit 7). Subtraction is included because it's executed by adding the two's complement. Also used by rotate instructions, which rotate the contents of a register and copy the low or high order bit of the register into the carry bit.

PIC16C84 Registers *(continued)*

Bit	Function
1	**Digit carry bit (DC).** Set if an addition or subtraction causes an overflow from the 4th low order bit (bit 3). Digit carry indicates that more than one hex digit (4 bits) was necessary to accommodate the result.
2	**Zero bit (Z).** Set if the result of an arithmetic or logic operation is zero.
3	**Power-down bit (PD).** Set during power-up or by a CLR WDT (clear watchdog) instruction. Cleared by a SLEEP instruction.
4	**Time-out bit (TO).** Set during power-up, by CLR WDT, or by SLEEP. Cleared by a watchdog timer time-out.
5-6	**Register page select bits for direct addressing (RP0, RP1).** These bits determine which register page is selected for direct addressing operations. Each page is 128 bytes long, so only RP0 is valid in a PIC16C71. RP1 can be used for storage, but may have an actual use in future PIC's.
7	**Register page select bit for indirect addressing (IRP).** This bit determines which register page is selected for indirect addressing operations. Since each indirect page is 256 bytes long, however, it is not useful in the PIC16C84. IRP can be used for storage, but may have an actual use in future PIC's.

PIC16C84 Registers *(continued)*

The following table shows how various events affect the power-down and time-out bits:

Event	PD	TO
Power-up	1	1
Watchdog time-out	x	0
SLEEP instruction	0	1
CLR WDT instruction	1	1

Lastly, this table shows the status of the power-down and time-out bits after a reset:

Cause of Reset	PD	TO
Watchdog time-out (not during sleep)	1	0
Watchdog time-out (during sleep)	0	0
External reset (not during sleep)	x	x
External reset (during sleep)	0	1
Normal power-up	1	1

- **File Select Register (04h: FSR).** This register selects the register for indirect addressing. Bits 0-7 select 1 of 256 registers in the current bank (the '84 only has one bank). As described in the PIC16C5x register descriptions, a read or write to register 00h will access the register pointed to by the FSR.
- **I/O Port A (05h).** 5-bit I/O port. This register is used to read and write I/O Port A. It can be read and written just as any other register. However, read instructions always read the I/O pins, regardless of whether the pins are programmed as inputs or outputs.

  Bit 4 (pin 3) has an open-collector output and shares its pin with the RTCC input.

PIC16C71 Registers *(continued)*

- **I/O Port B (06h).** 8-bit I/O port. Each of the Port B pins has a weak internal pull-up resistor (~250 µA). A pin's pull-up is turned off if the pin is configured as an output, and a single bit in the Option register can turn off all the pull-ups. The pull-up resistors are disabled on power-on reset.

  On bits 4-7 (pins 10-13), Port B has an interrupt on change feature that can generate an interrupt if any of the pins changes state. Any pin configured as an output is excluded from the interrupt feature.

  This interrupt can wake up the chip from sleep. Along with the internal pull-ups, the interrupt from sleep feature makes it easy to interface to a keypad and have wake-up on key press.

- **EEPROM Data Register (08h: EEDATA).** This register is used to read and write the 64 EEPROM registers.

- **EEPROM Address Register (09h: EEADR).** This register is used to set the address of the EEPROM register that will be read or written.

- **Program Counter Latch High (0Ah: PCLATH).** This register is used to access the 5 high bits of the program counter (PCH). Unlike the lower 8 bits (PCL), the 5 high bits are not directly addresable. Instead, they are stored in PCLATH for later use. When the program counter is loaded with a new value during a JMP, CALL, or a write to PCL, the high bits are loaded from PCLATH.

PIC16C84 Registers *(continued)*

- **Interrupt Control Register (0Bh: INTCON).** This register is used to enable interrupts and to determine what caused an interrupt. The function of each bit is given below:

Bit	Function
0	**Port B interrupt.** This bit is set if an interrupt was the result of a transition on any of the upper four bits of Port B. It will remain set until cleared by software.
1	**External interrupt.** Set if an interrupt was caused by a transition on the external INT pin. Must be cleared by software.
2	**RTCC overflow interrupt.** Set if an interrupt was caused by an overflow in the real-time clock/counter. Must be cleared by software.
3	**Port B interrupt enable.** Determines whether the Port B interrupt is enabled (“1”) or disabled (“0”).
4	**External interrupt enable.** Determines whether the external interrupt is enabled (“1”) or disabled (“0”).
5	**RTCC interrupt enable.** Determines whether the RTCC interrupt is enabled (“1”) or disabled (“0”).
6	**EEPROM interrupt enable.** Determines whether the EEPROM write complete interrupt is enabled (“1”) or disabled (“0”).
7	**Global interrupt enable.** Clearing this bit disables all interrupts. Setting this bit allows all interrupts that are individually enabled in bits 3-6.

- **Indirect Addressing Register (80h).** This register has the same function as register 00h.

PIC16C84 Registers *(continued)*

- **Option Register (81h: OPTION).** This register is used to set prescaler options, RTCC settings, external interrupt trigger edge, and Port B pull-up status. The function of each bit is shown below:

Bit	Function
0-2	**Prescaler ratio.** These 3 bits determine the prescaler input-to-output ratio. When using the prescaler with the RTCC, the seven possible ratios are 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, 1:128, 1:256. When using the prescaler with the watchdog timer, the ratios are 1:1, 1:2, 1:4, 1:8, 1:16, 1:32, 1:64, 1:128. For example, let's say that the prescaler is assigned to the watchdog timer. To increase the watchdog time-out period to 64 times its normal length, the prescaler ratio would be set to 110b; this yields a watchdog period of approx. 1 second (64 x 18 ms).
3	**Prescaler assignment.** This bit determines whether the prescaler is assigned to the RTCC ("0") or to the watchdog timer ("1").
4	**RTCC trigger edge.** This bit determines whether the RTCC increments on a low-to-high ("0") or high-to-low ("1") transition on the RTCC pin.
5	**RTCC signal source.** This bit determines whether the RTCC is driven by the PIC's internal instruction clock ("0") or by the RTCC pin ("1").
6	**External interrupt trigger edge.** Determines whether an interrupt will be caused by a high-to-low ("0") or low-to-high ("1") transition on the external INT pin.
7	**Port B pull-up status.** If this bit is clear, Port B pull-up resistors are enabled on pins that are inputs. If this bit is set, all Port B pull-ups are disabled.

PIC16C84 Registers *(continued)*

- **Program Counter Low Byte (82h: PCL).** Same as register 02h.
- **Status Register (83h).** Same as register 03h.
- **File Select Register (84h: FSR).** Same as register 04h.
- **Data Direction Register for Port A (85h: TRISA).** This is the data direction register for Port A. In the Microchip data book, this register is referred to as "Tri-State A", hence "TRISA" as an abbreviation.

  Bits in this register which are set to "1" cause the corresponding bits in Port A to become inputs (the pins go into high impedance mode, allowing them to be driven by an external source). Bits which are cleared to "0" cause the corresponding bits in Port A to become outputs.
- **Data Direction Register for Port B (86h: TRISB).** This is the data direction register for Port B.

  Data direction registers in the PIC16C84 are addressable, unlike their counterparts in PIC16C5x devices.
- **EEPROM Control Register 1 (88h: EECON1).** This register is used to control the reading and writing of the 64 EEPROM registers. It's also used to determine if the last write cycle was completed. The function of each bit is given below:

Bit	Function
0	**Read control bit.** Setting this bit starts an EEPROM read cycle. The bit is automatically cleared after the read.
1	**Write control bit.** Setting this bit starts an EEPROM write cycle. The bit is automatically cleared after the write.
2	**Write enable bit.** If this bit is set, the EEPROM can be written; if it's clear, the EEPROM can only be read.

PIC16C84 Registers *(continued)*

Bit	Function
3	**Write error flag.** This bit is automatically set if an EEPROM write cycle is prematurely terminated by an MCLR reset (during sleep or normal operation) or by a watchdog reset (during normal operation).
4	**Write completion interrupt flag.** Set when an EEPROM write cycle is completed. Must be cleared in software. Corresponding interrupt enable bit is in INTCON register.

- **EEPROM Control Register 2 (89h: EECON2).** This register is not physically implemented, and therefore cannot be read. However, EECON2 must be written to whenever an EEPROM write cycle is performed. See the EEPROM read / write notes following the register descriptions.
- **Program Counter Latch High (8Ah).** Same as register 0Ah.
- **Interrupt Control Register (8Bh: INTCON).** Same as register 0Bh.

Reading & Writing EEPROM Registers

Separate from the PIC16C84's 1K of program EEPROM space, there exist 64 bytes of EEPROM that can be used by your program as nonvolatile data storage.

This memory is not mapped in the normal register space. Instead, it is accessed through two registers: EEDATA (holds data to be read / written) and EEADR (holds address of EEPROM location). Additionally, two special registers control reading and writing of the EEPROM.

Reading from the EEPROM. To read an EEPROM location, your program must write the address to the EEADR register and then set bit 0 of EECON1 (read control bit). The desired data can be read from EEDATA in the next instruction. The following code shows how to read the EEPROM:

Reading & Writing EEPROM Registers *(continued)*

```
mov   EEADR,#00h   ;Select first EEPROM location
setb  STATUS.5     ;Select register page 1
setb  EECON1.0     ;Start read cycle
clrb  STATUS.5     ;Select register page 0
mov   w,EEDATA     ;Read data
```

Writing to the EEPROM. To write an EEPROM location, your program must write the address to the EEADR register and the data to the EEDATA register. Then your program must set bit 2 of EECON1 (write enable), write a sequence of two bytes to EECON2, and then set bit 1 of EECON1 (write control bit). The write operation takes approximately 10 ms. The following code shows how to write the EEPROM:

```
mov   EEADR,#00h   ;Select first EEPROM location
mov   EEDATA,#41h  ;Store ASCII value for 'A'
setb  STATUS.5     ;Select register page 1
setb  EECON1.2     ;Set write enable bit
mov   EECON2,#55h  ;Write special byte #1
mov   EECON2,#AAh  ;Write special byte #2
setb  EECON1.1     ;Start write cycle
jb    EECON1.1,$   ;Wait until write bit clears
clrb  STATUS.5     ;Select register page 0
```